To Rust or not to Rust

Alik Kurdyukov

High Load ++ Armenia

Co-organizer

Yandex

About me

- 20+ years in software develo
- UnitedTraders CTO
- Still write code (Kotlin, C#, Ru

Story plan

- Requirements
- General architecture
- Technology choice
- Rust path
- HOWTO

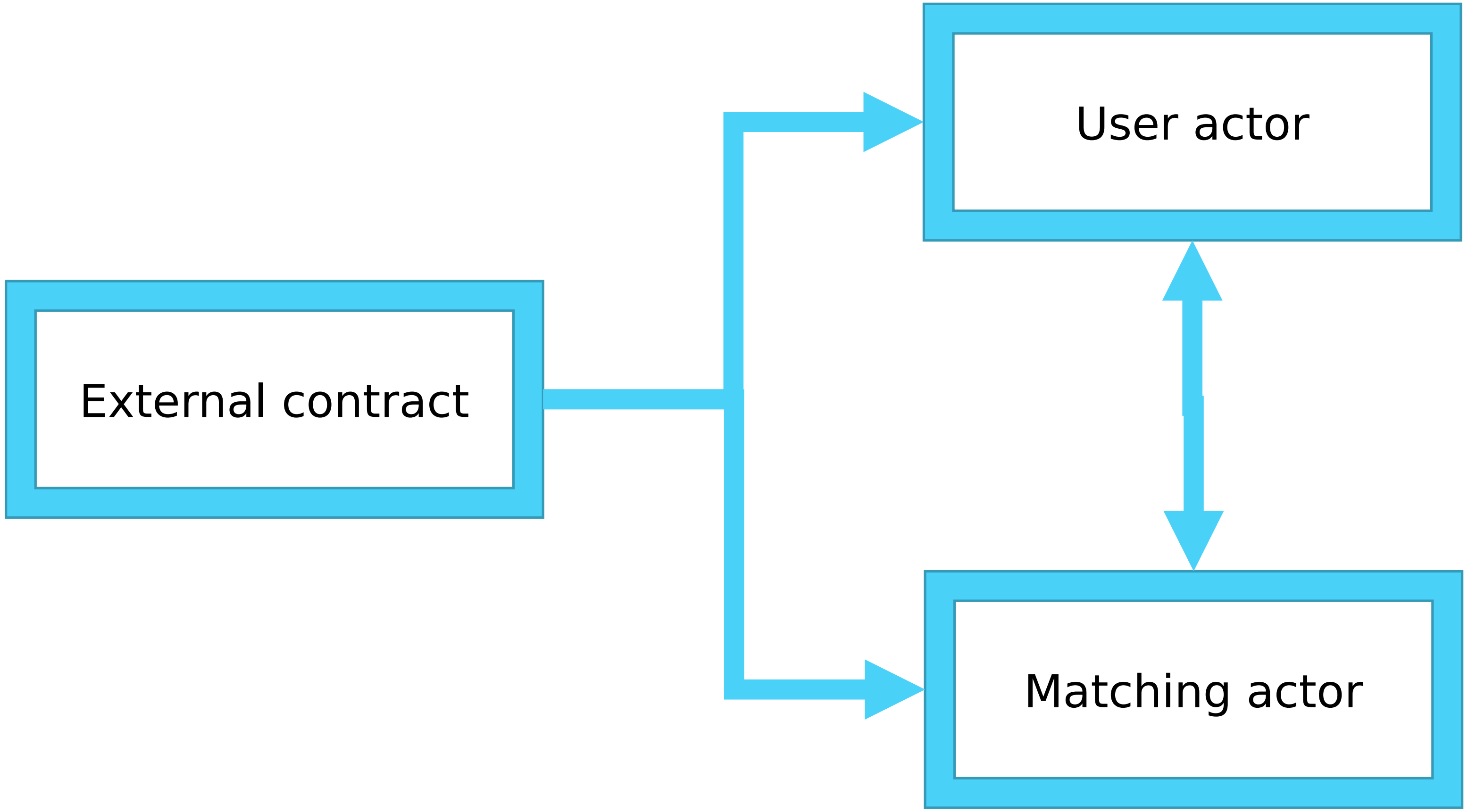
Requirements

- Matches sellers (ask) and buyers (buy)
- 24/7
- Under 1ms in 1000rps
- Loose nothing
- And many more…

Basic pipeline

1. Receive request
2. Validate incoming order
3. Reserve funds
4. Match
5. Un-reserve fund
6. Send response

General architecture

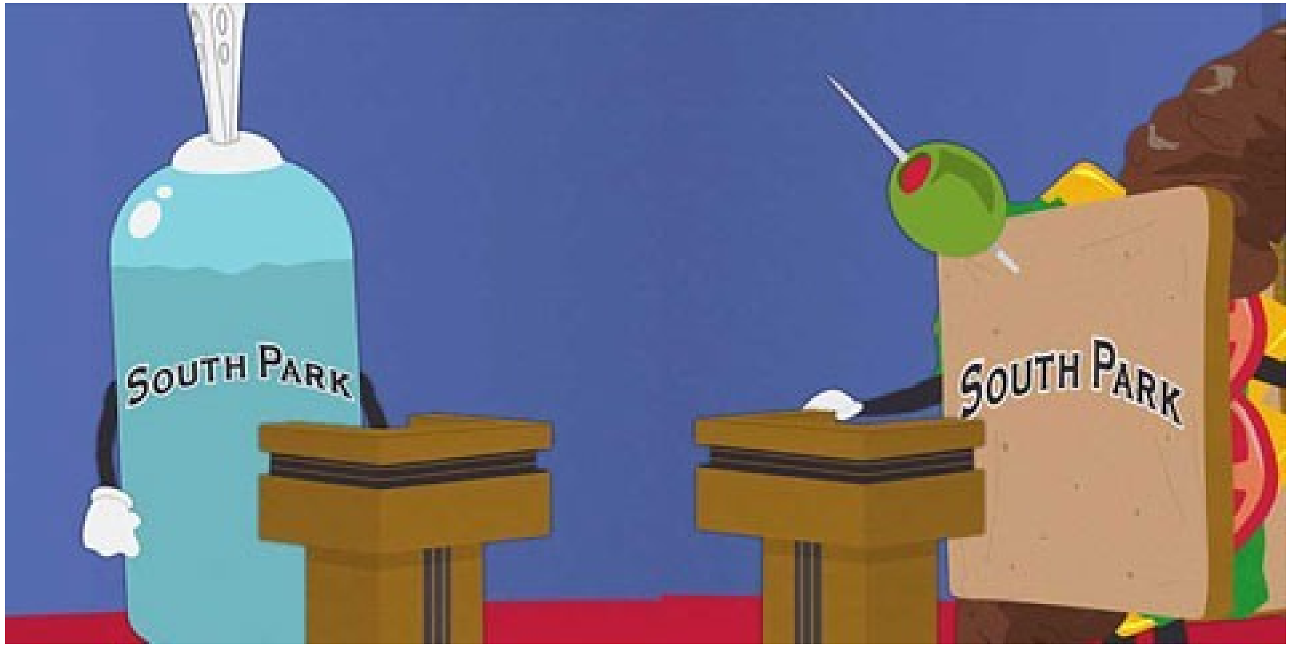

External contract

User actor

Matching actor

Technology choice

- GC: JVM, C#, Go, Erlan
- Non-GC: C++, Rust



Erlang

Pros:

- Native actors
- Performant GC

Cons:

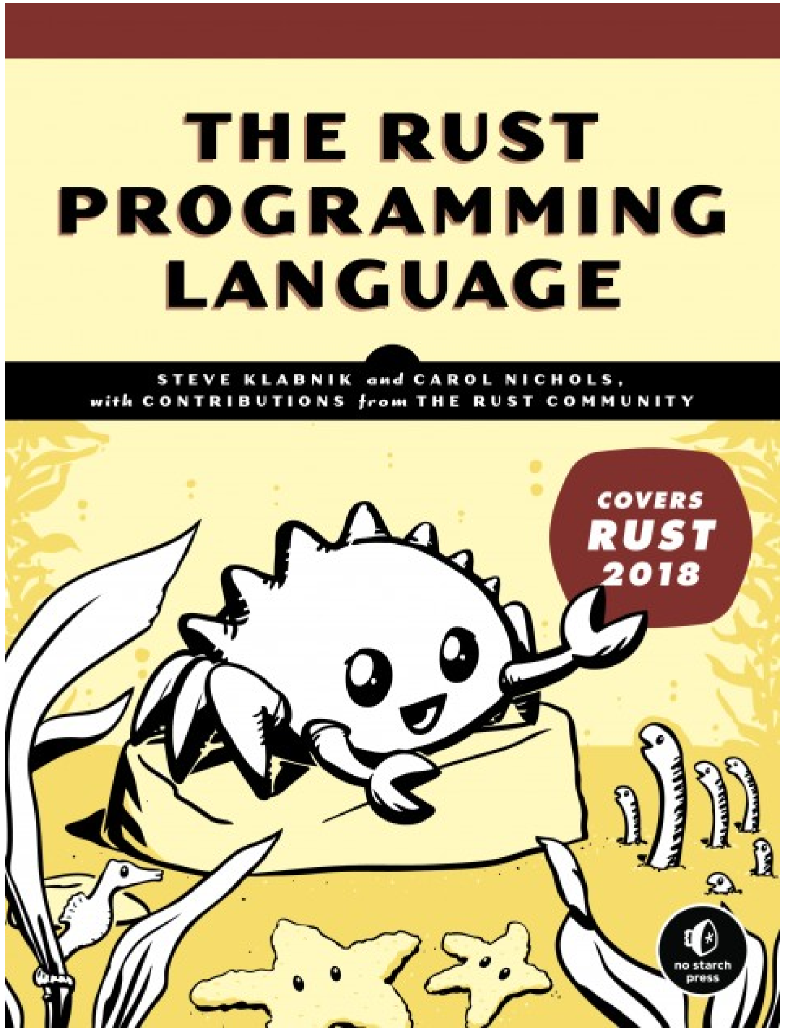
- Lack of engineers
- Steep learning curve
- Ugly syntax

Rust vs C++

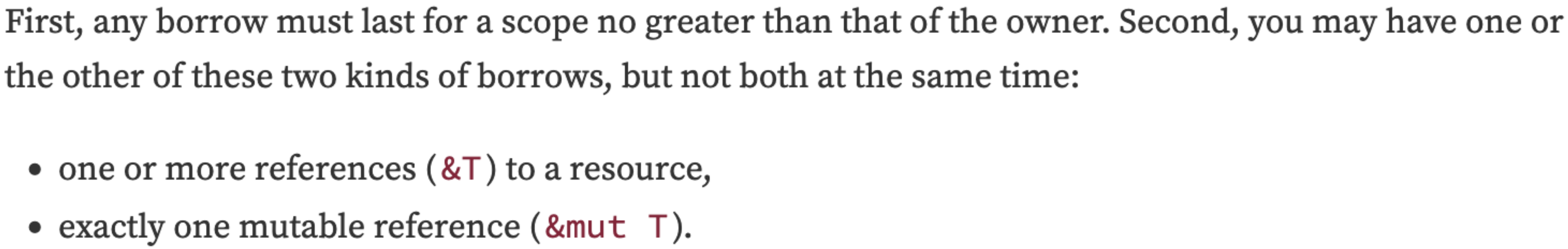
- Package manager in 20
- Open-Source Libraries
- HR

Rust Experiment

The Rust book


THE RUST PROGRAMMING LANGUAGE
STEVE KLABNIK and CAROL NICHOLS, with CONTRIBUTIONS from THE RUST COMMUNITY
COVERS RUST 2018
no starch press

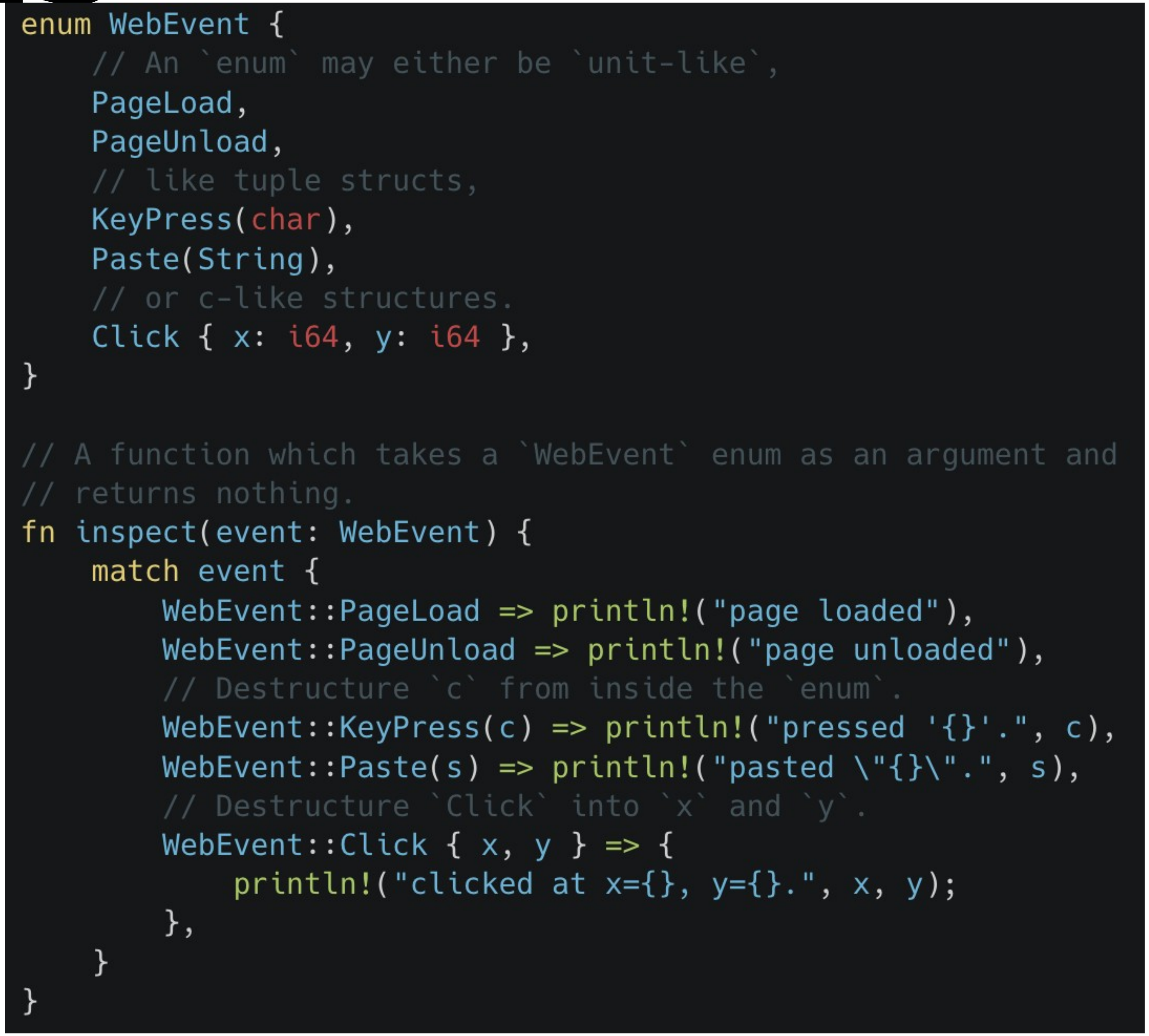
Borrow checker



First, any borrow must last for a scope no greater than that of the owner. Second, you may have one or the other of these two kinds of borrows, but not both at the same time:

- one or more references (`&T`) to a resource,
- exactly one mutable reference (`&mut T`).

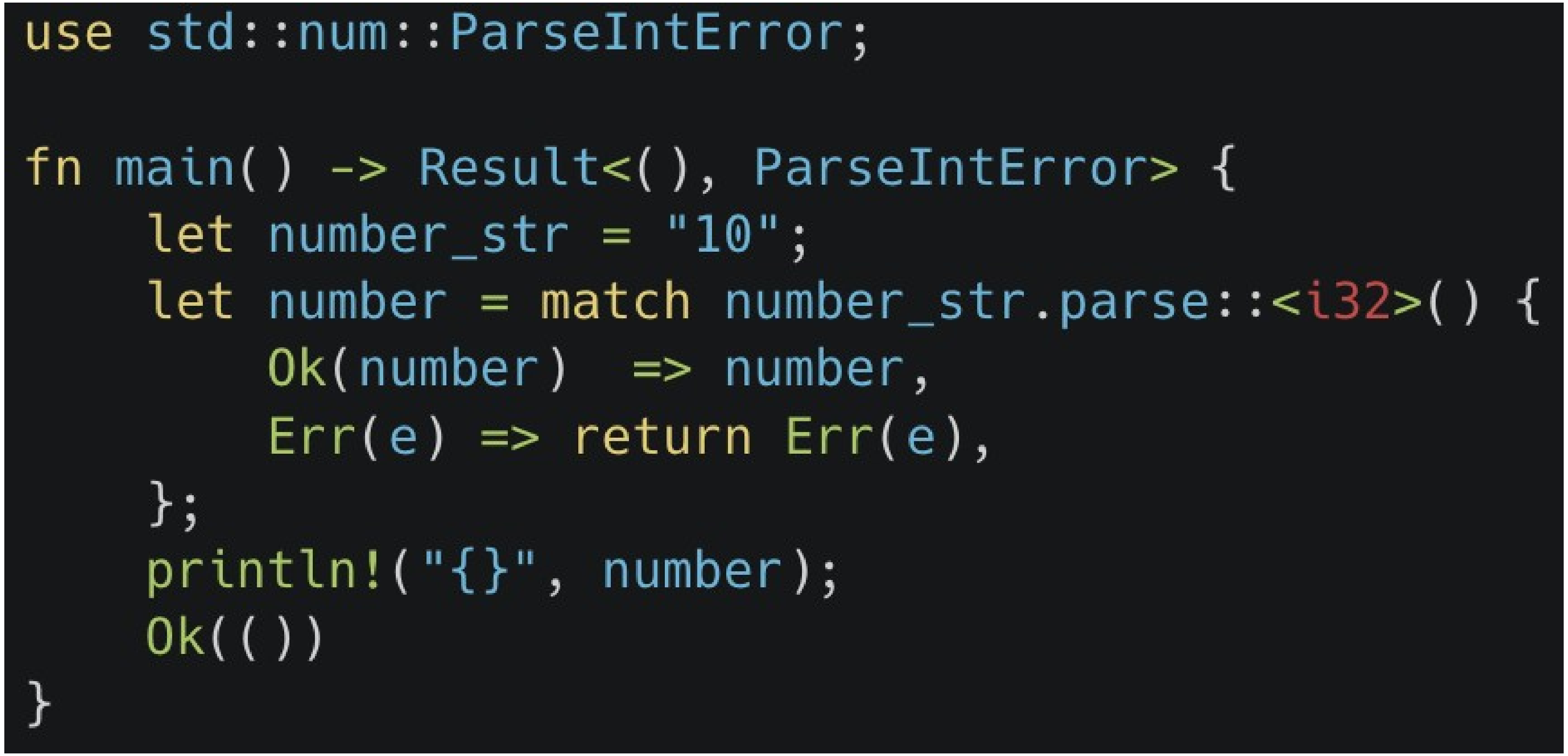
Enums



```rust
enum WebEvent {
    // An `enum` may either be `unit-like`,
    PageLoad,
    PageUnload,
    // like tuple structs,
    KeyPress(char),
    Paste(String),
    // or c-like structures.
    Click { x: i64, y: i64 },
}

// A function which takes a `WebEvent` enum as an argument and
// returns nothing.
fn inspect(event: WebEvent) {
    match event {
        WebEvent::PageLoad => println!("page loaded"),
        WebEvent::PageUnload => println!("page unloaded"),
        // Destructure `c` from inside the `enum`.
        WebEvent::KeyPress(c) => println!("pressed '{}'.", c),
        WebEvent::Paste(s) => println!("pasted \"{}\".", s),
        // Destructure `Click` into `x` and `y`.
        WebEvent::Click { x, y } => {
            println!("clicked at x={}, y={}.", x, y);
        },
    }
}
```

Error handling

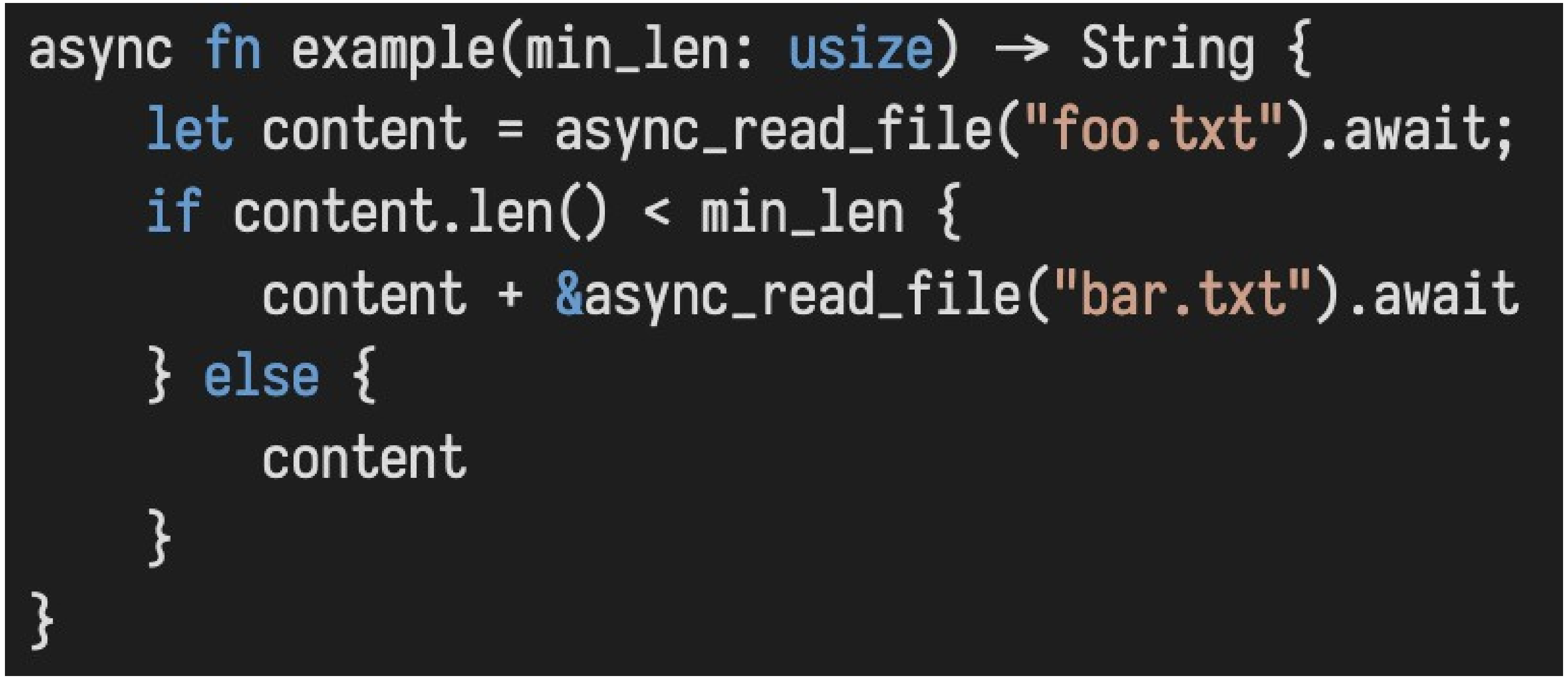


```
use std::num::ParseIntError;

fn main() -> Result<(), ParseIntError> {
    let number_str = "10";
    let number = match number_str.parse::<i32>() {
        Ok(number)  => number,
        Err(e) => return Err(e),
    };
    println!("{}", number);
    Ok(())
}
```

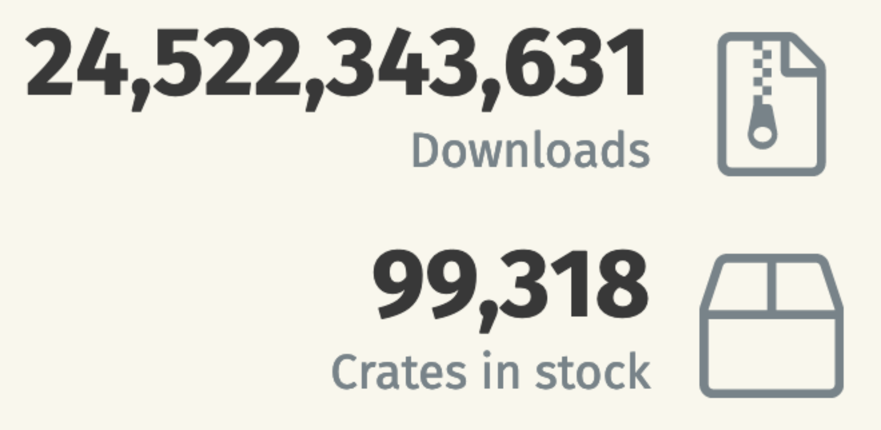
Zero cost concurrency

- Polling Futures
- Async/await



```rust
async fn example(min_len: usize) -> String {
    let content = async_read_file("foo.txt").await;
    if content.len() < min_len {
        content + &async_read_file("bar.txt").await
    } else {
        content
    }
}
```

Libraries



24,522,343,631

Downloads

99,318

Crates in stock

Basic scenarios

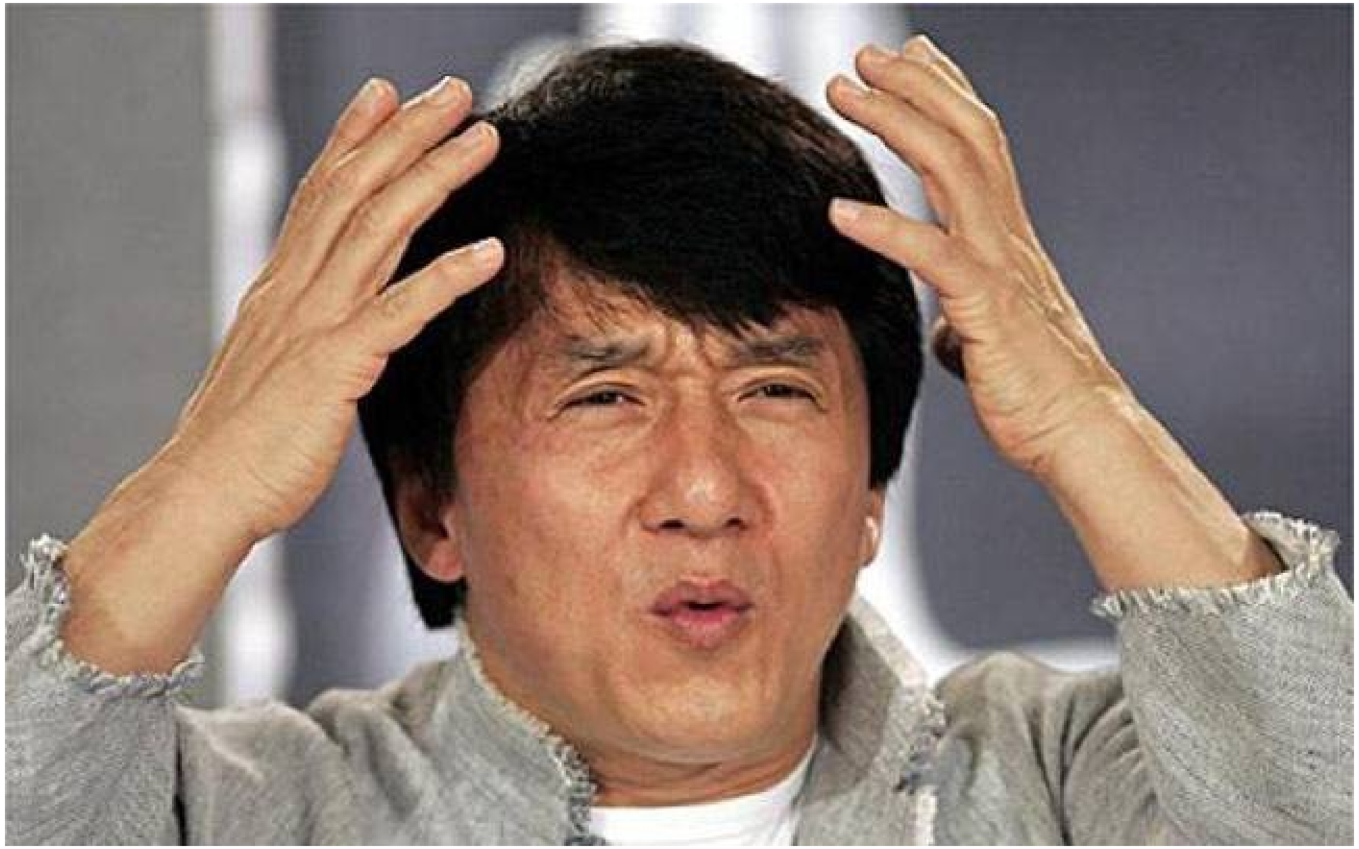
- DDD
- Tests
- 2 weeks
- 1.5x with RocksDB storage vs Erlang without one

Growing Team

- Migration
- Enthusiasm

Business expectations

- Progress report
- Risk model



Aeron port

- Only C++ API
- Idiomatic port

Problems solved

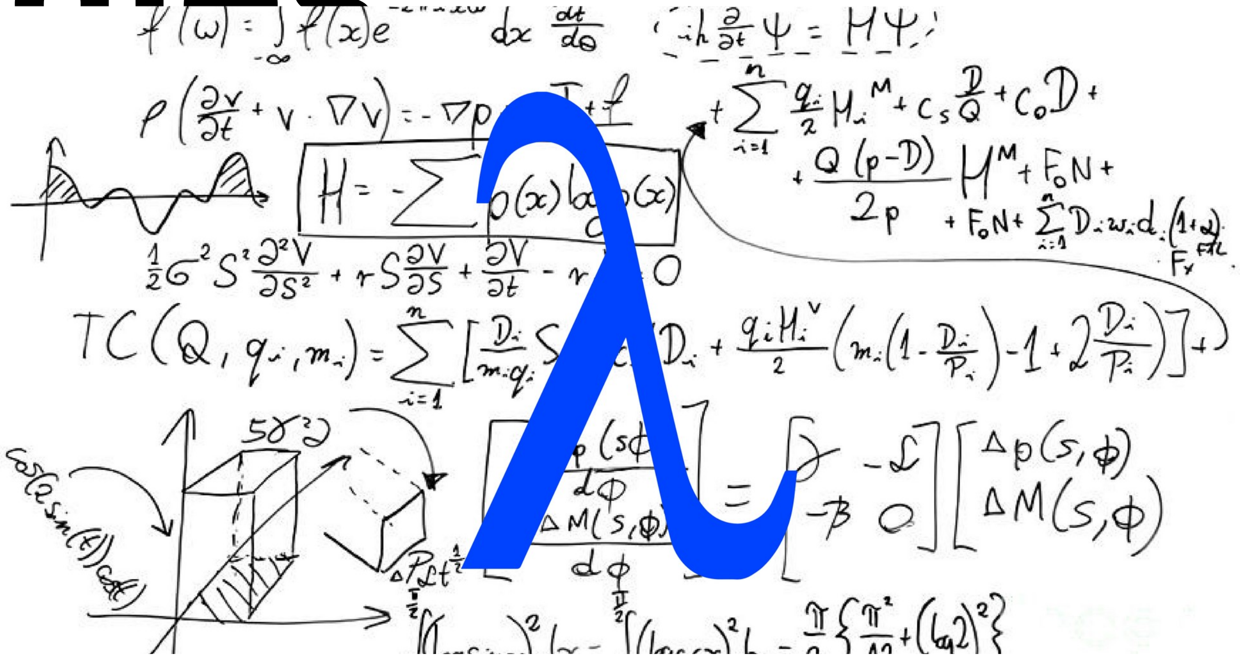
- Unstable Futures
- Libraries upgrade

HOW
TO

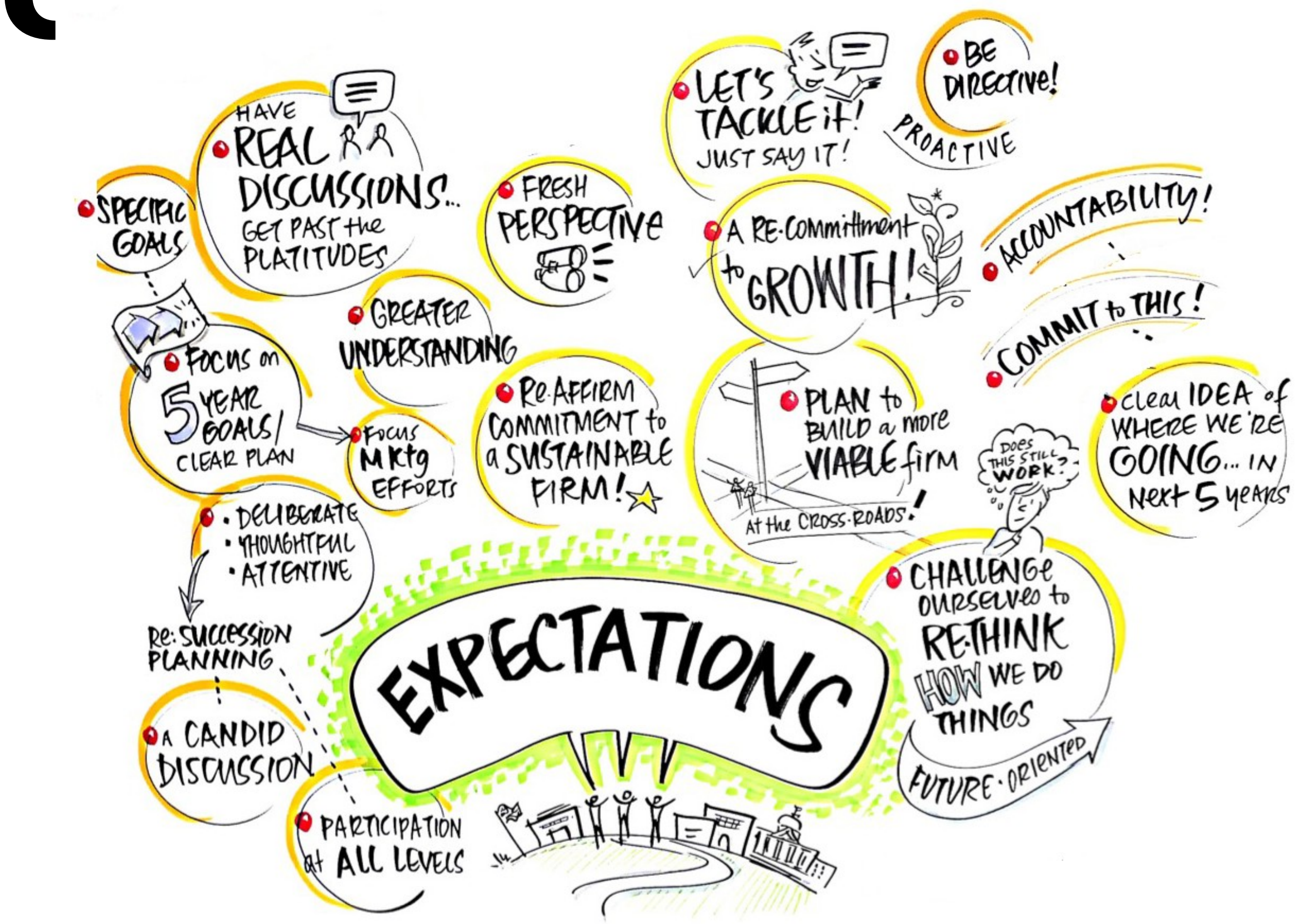
Start with a team

Check ecosystem

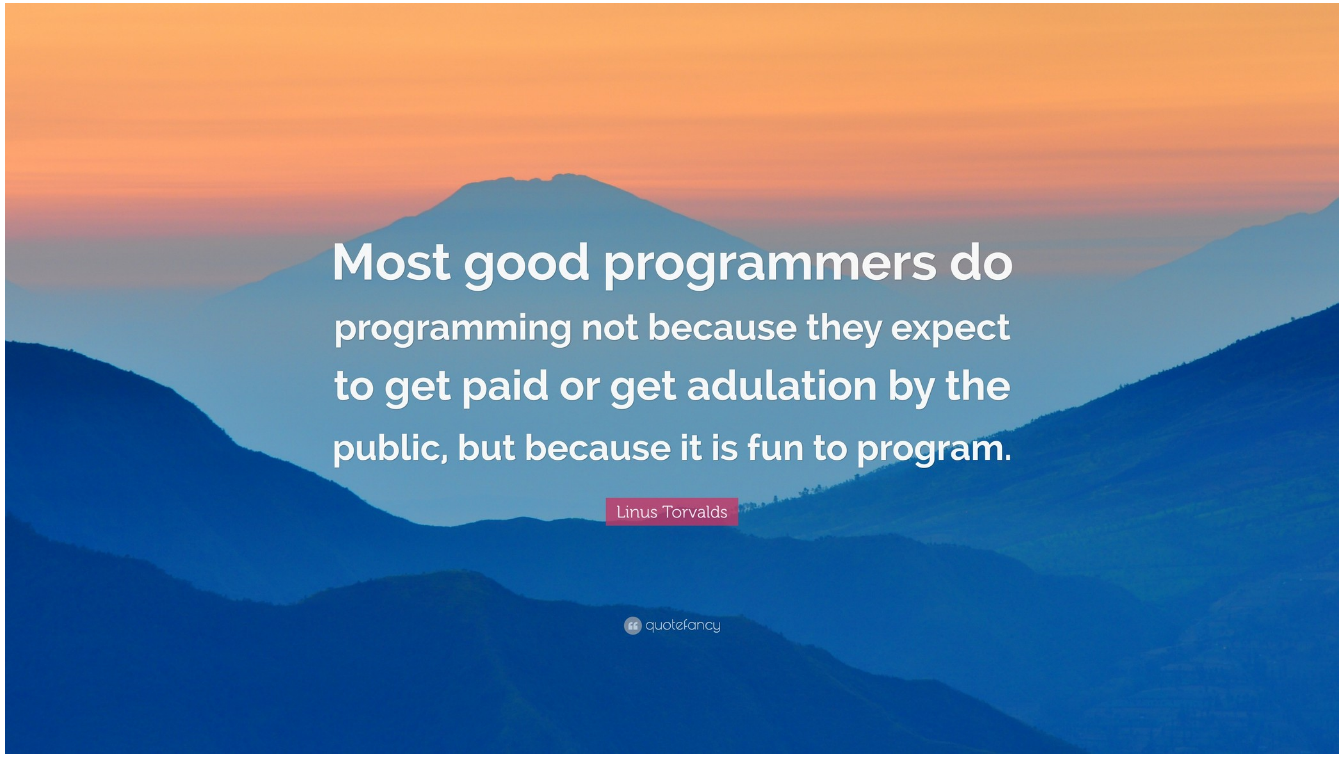
Prepare to change strategies



Manage business expectations


HAVE REAL DISCUSSIONS... GET PAST the PLATITUDES
SPECIFIC GOALS
FRESH PERSPECTIVE
LET'S TACKLE it! JUST SAY IT!
BE DIRECTIVE!
PROACTIVE
A RE-COMMITMENT to GROWTH!
ACCOUNTABILITY!
COMMIT to THIS!
GREATER UNDERSTANDING
FOCUS on 5 YEAR GOALS/ CLEAR PLAN
FOCUS MKtg EFFORTS
Re-AFFIRM COMMITMENT to a SUSTAINABLE FIRM!
PLAN to BUILD a more VIABLE firm!
At the CROSS-ROADS!
DOES THIS STILL WORK?
clear IDEA of WHERE WE'RE GOING... IN Next 5 years
• DELIBERATE • THOUGHTFUL • ATTENTIVE
Re: SUCCESSION PLANNING
EXPECTATIONS
CHALLENGE OURSELVES to RE-THINK HOW WE DO THINGS
FUTURE-ORIENTED
A CANDID DISCUSSION
PARTICIPATION at ALL LEVELS

Have fun!



Most good programmers do programming not because they expect to get paid or get adulation by the public, but because it is fun to program.

Linus Torvalds

quotefancy

Thanks! Questions?

Please rate the talk ->

++ High Load Armenia

Co-organizer

Yandex